AMENDMENTS TO THE CLAIMS

1. (ORIGINAL) A method of emulating IP devices in a protocol emulator, the method comprising:

promiscuously detecting IP frames on a network interface;

generating, for each corresponding detected IP frame, a response IP frame by an executable emulation application; and

outputting each said response IP frame by a raw socket onto the network interface.

2. (CURRENTLY AMENDED) The method of claim 1, wherein the protocol emulator has a prescribed assigned IP address and a kernel configured for detecting IP frames, the promiscuously detecting step including one of: (1) passing by the kernel the detected IP frames to the executable emulation application independent of any specified IP destination address, including any IP destination addresses different from the prescribed assigned IP address, [[and]] or (2) monitoring by the executable emulation application for the detected IP frames received by the network interface.

3. (ORIGINAL) The method of claim 2, wherein the generating step includes:

determining from each detected IP frame the corresponding supplied IP source address and supplied IP destination address;

generating an IP header for the corresponding response IP frame having a response IP source address and a response IP destination address, the response IP source address having a value corresponding to the supplied IP destination address and the response IP destination address having a value corresponding to the supplied IP source address.

4. (ORIGINAL) The method of claim 3, wherein the generating step further includes generating application response data based on parsed application request data within the corresponding detected IP frame.

5. (ORIGINAL) The method of claim 4, wherein the parsed application request data includes a hypertext transfer protocol (HTTP) get request, the step of generating application response data including providing HTML page data in the corresponding response IP frame based on the HTTP get request.

6. (ORIGINAL) The method of claim 3, wherein the generating step further includes generating, for the corresponding response IP frame, layer 4 header information by the executable emulation application.

7. (ORIGINAL) The method of claim 3, wherein the generating step further includes:

identifying the parsed application request data by one of a plurality of available executable emulation applications; and

generating the application response data and the response IP frame for the parsed application request data by the one available executable emulation application.

8. (PREVIOUSLY PRESENTED) The method of claim 2, wherein the promiscuously detecting includes monitoring by the executable emulation application for an identifiable application request within the detected IP frames, the executable emulation application generating application response data for the response IP frame based on the identifiable application request.

9. (PREVIOUSLY PRESENTED) The method of claim 2, wherein the promiscuously detecting includes monitoring by a plurality of available executable applications for respective prescribed application requests within the detected IP frames.

10. (ORIGINAL) The method of claim 9, wherein the generating step includes generating application response data for the response IP frame by one of the available executable applications having identified the corresponding prescribed application request.

11. (PREVIOUSLY PRESENTED) The method of claim 2, wherein the passing step includes configuring the kernel to pass the detected IP frames having the destination addresses different from the prescribed assigned IP address.

12. (ORIGINAL) An emulator comprising:

a network interface configured for receiving IP frames from a network, each received IP frame having a corresponding IP source address and a corresponding IP destination address;

an executable emulation application configured for promiscuously detecting the IP frames received by the network interface, the executable emulation application configured for generating a response IP frame for each corresponding detected IP frame having a corresponding identified application request, independent of the corresponding IP source address within the detected IP frame; and

a raw socket configured for outputting the response IP frame to the network interface for transmission on the network.

13. (PREVIOUSLY PRESENTED) The emulator of claim 12, further comprising a kernel configured for passing the detected IP frames from the network interface toward the executable emulation application independent of any specified IP destination addresses.

14. (ORIGINAL) The emulator of claim 12, wherein the executable emulation application is configured for generating the response IP frame by generating a response IP source address and a response IP destination address, the response IP source address having a value matching the IP destination address of the corresponding detected IP frame, and the response IP

destination address having a value matching the IP source address of the corresponding detected IP frame.

15. (ORIGINAL) The emulator of claim 14, wherein the executable emulation application is further configured for generating application response data based on parsed application request data within the corresponding detected IP frame.

16. (ORIGINAL) The emulator of claim 15, wherein the parsed application request data includes a hypertext transfer protocol (HTTP) get request, the executable emulation application configured for generating, within the application response data, HTML page data based on the HTTP get request.

17. (ORIGINAL) The emulator of claim 14, wherein the executable emulation application generates layer 4 header information for the response IP frame.

18. (ORIGINAL) The emulator of claim 12, further comprising a plurality of available executable emulation applications, each configured for generating a corresponding response IP frame for said each corresponding detected IP frame based on the corresponding identified application request, independent of the corresponding IP source addresses within the detected IP frame, wherein the response IP frame for each corresponding detected IP frame is generated by a corresponding one of the available executable emulation applications.

19. (ORIGINAL) A computer readable medium having stored thereon sequences of instructions for emulating IP devices in a protocol emulator, the sequences of instructions including instructions for performing the steps of:

promiscuously detecting IP frames on a network interface;

generating, for each corresponding detected IP frame, a response IP frame by an executable emulation application; and

outputting each said response IP frame by a raw socket onto the network interface.

20. (CURRENTLY AMENDED) The medium of claim 19, wherein the protocol emulator has a prescribed assigned IP address and a kernel configured for detecting IP frames, the promiscuously detecting step including one of: (1) passing by the kernel the detected IP frames to the executable emulation application independent of any specified IP destination address, including any IP destination addresses different from the prescribed assigned IP address, [[and]] or (2) monitoring by the executable emulation application for the detected IP frames received by the network interface.

21. (ORIGINAL) The medium of claim 20, wherein the generating step includes:

determining from each detected IP frame the corresponding supplied IP source address and supplied IP destination address;

generating an IP header for the corresponding response IP frame having a response IP source address and a response IP destination address, the response IP source address having a value corresponding to the supplied IP destination address and the response IP destination address having a value corresponding to the supplied IP source address.

22. (ORIGINAL) The medium of claim 21, wherein the generating step further includes generating application response data based on parsed application request data within the corresponding detected IP frame.

23. (CANCELED).

24. (ORIGINAL) The medium of claim 21, wherein the generating step further includes generating, for the corresponding response IP frame, layer 4 header information by the executable emulation application.

25. (ORIGINAL) The medium of claim 21, wherein the generating step further includes:

identifying the parsed application request data by one of a plurality of available executable emulation applications; and

generating the application response data and the response IP frame for the parsed application request data by the one available executable emulation application.

26. (PREVIOUSLY PRESENTED) The medium of claim 20, wherein the promiscuously detecting includes monitoring by the executable emulation application for an identifiable application request within the detected IP frames, the executable emulation application generating application response data for the response IP frame based on the identifiable application request.

27. (PREVIOUSLY PRESENTED) The medium of claim 20, wherein the promiscuously detecting includes monitoring by a plurality of available executable applications for respective prescribed application requests within the detected IP frames.

28. (CANCELED).

29. (PREVIOUSLY PRESENTED) The medium of claim 20, wherein the passing step includes configuring the kernel to pass the detected IP frames having the destination addresses different from the prescribed assigned IP address.

30. (ORIGINAL) An emulator system comprising:

means for promiscuously detecting IP frames on a network interface;

means for generating, for each corresponding detected IP frame, a response IP frame by an executable emulation application; and

means for outputting each said response IP frame by a raw socket onto the network interface.

31. (PREVIOUSLY PRESENTED) The system of claim 30, wherein the emulator system has a prescribed assigned IP address and a kernel configured for detecting IP frames based on an IP source address, the promiscuously detecting means configured for passing the detected IP frames independent of any specified IP destination address, including any IP destination addresses different from the prescribed assigned IP address.

32. (ORIGINAL) The system of claim 31, wherein the generating means includes:

means for determining from each detected IP frame the corresponding supplied IP source address and supplied IP destination address;

means for generating an IP header for the corresponding response IP frame having a response IP source address and a response IP destination address, the response IP source address having a value corresponding to the supplied IP destination address and the response IP destination address having a value corresponding to the supplied IP source address.

33. (ORIGINAL) The system of claim 32, wherein the generating means further is configured for generating application response data based on parsed application request data within the corresponding detected IP frame.

34. (CANCELED).

35. (ORIGINAL) The system of claim 32, wherein the generating means is configured for generating, for the corresponding response IP frame, layer 4 header information by the executable emulation application.

36. (ORIGINAL) The system of claim 32, wherein the generating means is configured for:

identifying the parsed application request data by one of a plurality of available executable emulation applications; and

generating the application response data and the response IP frame for the parsed application request data by the one available executable emulation application.

37. (ORIGINAL) The system of claim 31, wherein the generating means monitors for an identifiable application request within the detected IP frames, and generates the application response data for the response IP frame based on the identifiable application request.

38. (ORIGINAL) The system of claim 31, wherein the generating means includes a plurality of available executable applications for monitoring respective prescribed application requests within the detected IP frames.

39. (ORIGINAL) The system of claim 38, wherein the generating means is configured for generating application response data for the response IP frame by one of the available executable applications having identified the corresponding prescribed application request.

40. (PREVIOUSLY PRESENTED) The method of claim 1, wherein the promiscuously detecting, generating, and outputting each are performed in the protocol emulator.

41. (PREVIOUSLY PRESENTED) The emulator of claim 12, wherein the executable emulation application is configured for promiscuously detecting the detected IP frames by monitoring for the received IP frames received by the network interface.

42. (PREVIOUSLY PRESENTED) The medium of claim 19, wherein the promiscuously detecting, generating, and outputting each are performed in the protocol emulator.